Docket No.: NS-3799 US

LAW OFFICES OF
SKJERVEN, MORRILL, MacPHERSON, FRANKLIN & FRIEL LLP
25 METRO DRIVE, SUITE 700
San Jose, California 95110
(408) 453-9200

OTHER OFFICES:
AUSTIN, TEXAS
NEWPORT BEACH, CALIFORNIA
SAN FRANCISCO, CALIFORNIA

FACSIMILE (408) 453-7979

09/301438 U.S. PTO 04/28/99

April 28, 1999

09/301438 U.S. PTO jc511 04/28/99

Box Patent Application
ASSISTANT COMMISSIONER FOR PATENTS
Washington, D. C. 20231

Enclosed herewith for filing is a patent application, as follows:

Inventors: Christopher K. Wolf, Ygal Arbel, Himanshu A. Sanghavi
Title: DVD/DVB Decoder

9 Sheets of Drawings
28 pages Specification
8 pages Claims
2 page Abstract
3 pages signed Declaration/Power of Attorney
1 pages Recordation Cover Sheet (in duplicate)
2 pages Assignment
__ page(s) PTO Form 1449
☐ Copy(ies) of __ Cited References
☐ Other: _____

CLAIMS AS FILED (fees computed under §1.9(f))

For	Number Filed			Number Extra		Rate			Basic Fee 760.00
Total Claims	38	-20	=	18	x	$18	=	$	324.00
Independent Claims	2	-3	=	0	x	$78	=	$	0.00
☐ Application contains one or more multiple dependent claims ($260 total fee)								$	0.00

Please make the following charges to Deposit Account 19-2386:

☒ Fee for filing the patent application in the amount of $1,084.00

☒ The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account 19-2386.

A Return Post Card and this sheet in duplicate are enclosed.

EXPRESS MAIL LABEL NO:
EM541633669US

Respectfully submitted,

Edward C. Kwok
Attorney for Applicant(s)
Reg. No. 33,938

PATENT APPLICATION

EXPRESS MAIL LABEL NO:
EM541633669US

DVD/DVB DECODER

Christopher K. Wolf
Ygal Arbel
Himanshu A. Sanghavi

FIELD OF THE INVENTION

The present invention relates to a decoder system and, more particularly, to a decoder system for decoding Digital Versatile Disc (DVD) or Digital Video Broadcast (DVB) data streams.

0930150-050303

BACKGROUND

Digital Versatile Disc (DVD) and Digital Video Broadcast (DVB) data decoders are well known. A conventional DVD/DVB decoder system includes a central processing unit, a stream demultiplexer, an audio decoder, a video decoder, a memory controller, an MPEG system timer, a video output processor and an audio output processor.

Conventional DVD/DVB decoder systems have many disadvantages. First, they typically include two separate depacketizers; an audio depacketizer coupled to the audio decoder for depacketizing audio data, and a video depacketizer coupled to the video decoder for depacketizing video data, leading to duplication of many tasks. Therefore, such systems are not cost effective.

Second, the Central Processing Unit (CPU) of conventional DVD/DVB systems is frequently interrupted,

requiring the CPU to temporarily abandon its existing tasks to service the interrupts, leading to inefficient utilization of the CPU. Furthermore, because the interrupts are frequent and not synchronized to a common event, such systems must meet strict and inflexible timing requirements. Moreover, in such systems, because the CPU typically demultiplexes the stream data in addition to handling many other tasks, it must be very powerful and hence relatively expensive.

Third, the Packetized Elementary Stream (PES) data supplied to the audio and the video decoders of conventional DVD/DVB systems typically includes a time stamp which must be stripped off before the data is decoded, resulting in processing inefficiencies.

Docket No. 50T5010.US

SUMMARY

The Digital Versatile Disc (DVD) and Digital Video Broadcast (DVB) decoder, in accordance with one embodiment of the present invention, includes a stream demultiplexer, a data storage buffer and a control Central Processing Unit (CPU). The stream demultiplexer demultiplexes and depacketizes the raw DVD data stream, subsequently stores the video and audio data in the storage buffer and informs the CPU thereabout.

The stream demultiplexer extracts the timing information embedded in each data pack of a data stream and, accordingly, generates a tag containing the decode time stamp, the presentation time stamp and the storage location of each data pack stored in the buffer.

The CPU reads the information recorded in each tag and, accordingly, generates task definition packets for use by the decoder.

The decoder is fully synchronized with respect to a signal generated by a video output processor of the system. Accordingly, in a steady state and under normal operating conditions, the CPU is interrupted only when the synchronization signal arrives thereby significantly reducing the number of interrupts that the CPU must service. Furthermore, the CPU benefits from having an entire period of the synchronization signal to service the interrupts and therefore has relaxed timing requirements.

The decoder is pipelined to reduce the idle time and hence to increase the utilization of the CPU and the video decoder. Accordingly, during each cycle of the synchronization signal, while the video decoder decodes the data, the CPU generates a task definition packet for the data to be decoded during the next cycle.

Because the stream demultiplexer removes all the timing information from the data before storing them in the storage buffer, the tasks performed by the audio and the video decoders are simplified. Furthermore, because the CPU is the component that makes all the decisions about the particular data and their decode time, the audio and video decoders have vastly simplified tasks.

The stream demultiplexer also depacketizes the audio data stream. The stream demultiplexer extracts

the timing information from each audio data pack and stores its payload in the buffer. Thereafter, the audio decoder determines the offset between the beginning of an audio data packet and the beginning of an audio frame and supplies the information to the CPU. The CPU using the time stamp information provided by the stream demultiplexer and the offset information provided by the audio decoder determines the presentation time of an audio frame.

Other aspects and advantages of the present invention are apparent from the following detailed description and accompanying drawings.

DOCUMENT OPENED

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 shows a block diagram of a decoder in accordance with one embodiment of the present invention.

Fig. 2A shows the various components of a DVD data pack.

Fig. 2B shows the various components of a packet of data pack of Fig. 2A.

Fig. 2C shows the various components of a DVB transport data packet.

Fig. 3 shows the basic components of a task definition packet in accordance with one embodiment of the present invention.

Fig. 4 shows the format of a message generated by the audio decoder to the CPU when a sync word in the audio stream is found.

Fig. 5 shows an audio buffer in accordance with

one embodiment of the present invention.

Fig. 6 shows the connections of the stream demultiplexer to various modules in the decoder of Fig. 1 in accordance with one embodiment of the present invention.

Fig. 7 shows an event tag format in accordance with one embodiment of the present invention.

Fig. 8 shows a DVD-oriented layered-packetized protocol that is transported by the stream demultiplexer of Fig. 1 in accordance with one embodiment of the present invention.

Fig. 9 shows a message queue in accordance with one embodiment of the present invention.

Docket No. 50T5003.001

DETAILED DESCRIPTION

A schematic block diagram of a Digital Versatile Disc (DVD)/Digital Video Broadcast (DVB) decoder 20, in accordance with one embodiment of the present invention, is illustrated in Fig. 1.

Decoder 20 uses three data paths in either the DVD or DVB mode of operation, namely a video data path, an audio data path, and a control path.

Decoder 20 includes, in addition to other components not shown, a Network Port (NP) 22, a DVD-DSP interface 24, a Stream Demultiplexer (SD) 26, a timer 30, a clock generator 32, an audio decoder 34, a video decoder 36, an Audio Output Processor (AOP) 38, a Video Output Processor (VOP) 40, a control Central Processing Unit (CPU) 54 and a register 56.

In decoder 20, only a subset of the above

components process the audio data. For example, audio decoder 38 only decodes encoded audio data. Similarly, the processing of video data is performed by a subset of the decoder 20 components. The processing of video data when decoder 20 is in the DVD mode of operation is described first.

The video data path of decoder 20 decodes and displays MPEG (MPEG1 or MPEG2) (developed by the Moving Pictures Expert Group of the International Standards Organization (ISO)) compressed video data synchronously with timer 30.

As shown in Fig. 1, the video data path includes DVD-DSP interface 24, SD 26, timer 30, video decoder 36, VOP 40, CPU 54 and register 56. In particular, DVD-DSP interface 24 retrieves raw DVD data bit streams and reformats it into bytes before supplying it to SD 26. SD 26 demultiplexes and depacketizes the video data stream and transports compressed video data to the compressed video buffer in buffer 48. Video decoder 36 fetches the compressed video data, decodes the data and stores the decoded data in one of the frame stores in buffer 50. VOP 40 retrieves the video frame data from buffer 50, merges the video frame data with OSD (on-screen display) and/or sub-picture data, and encodes the video frame data as an NTSC or PAL output for NTSC/PAL TV monitor 42.

In particular, depending on the data consumption rate, DVD-DSP Interface 24 retrieves raw data residing on DVD-DSP 46 at a typical rate of approximately 11 Megabit per second. The DVD data stream supplied to SD

26 contains compressed audio, video, control and synchronization information. SD 26 demultiplexes and depacketizes the data stream, storing the demultiplexed compressed audio and video data in data buffer 48 which is typically a First-In-First-Out (FIFO) buffer. SD 26 has access to 32 different addressable locations in FIFO 48. A host programmable memory, called the routing table (not shown in Fig. 1), directs SD 26 to a particular location within FIFO 48 in which the elementary streams and substreams of data are stored.

Each DVD data is composed of data units, commonly referred to as data packs. As illustrated in Fig. 2A, each DVD data pack 200 contains a pack header 201 and may contain one or more of the following elements: a system header 202, an audio Packetized Elementary Stream (PES) 204, a video PES packet and a control packet 208. As shown in Fig. 2B, each video PES packet 206 further contains a header 216 and a payload 214. Each header 216 contains a PES control field 218 and may contain a Decode Time Stamp (DTS) 210 or a Presentation Time Stamp (PTS) 212.

System header 202 identifies all the elementary streams of a pack. SD 26 places the entire system header 202 in its own buffer for access by CPU 54.

SD 26 first demultiplexes the audio, video and control components of the received DVD data. Thereafter, SD 26 depacketizes the demultiplexed data, namely, SD 26 removes header 208, DTS 210 and PTS 212 from each packet 206 before storing the payload 214 in buffer 48. Therefore, audio decoder 34 and video

decoder 36 receive only the payload 214 of data packet 206, which is an advantage of decoder 20. Because SD 26 demultiplexes the DVD data byte streams, several advantages are realized. First, CPU 54 is allowed to specialize in more complex and decision-based tasks. Second, unlike the control CPUs of conventional DVD decoders, CPU 54 is not required to have a high processing capability and is thus relatively inexpensive.

Fig. 2C shows the various components of a DVB transport data packet 230. Each DVB data packet 230 contains transport packet control 232, program identifier (PID) 234, adaptation field control 238, program clock reference (PCR) 236 and payload 240. Transport packet control 232 and PID 234 together form a transport packet header for the DVB data packet 230.

Concurrent references are made below to Figures 1-3 below. Each data pack 200, contains a timing data, embedded in its pack header 201, which establishes a timing reference with respect to which the PTS 212 of the associated PES packet 206 is measured. In other words, the display time of a payload 214 is determined by comparing its PTS 212 to the reference timing data embedded in the associated system header 202.

Timer 30 is a real time clock that establishes the time base for all the timing references in system header 202. Therefore, timer 30 detects whether the recorded times in system header 202 have arrived. Timer 30 also includes facilities which trigger events when a particular time is reached. Consequently, timer 30

generates events based on specific times or records the times when specific events occur.

As is seen from Fig. 1, Video Output Processor (VOP) 40 generates a signal, called Vsync, every 16 milliseconds (i.e. 60 HZ for NTSC systems) which is the typical rate at which a Cathode Ray Tube (CRT) or a TV monitor is refreshed. Signal Vsync is supplied to Central Processing Unit (CPU) 54, video decoder 36 and timer 30. Signal Vsync synchronizes the activities of the various components of decoder 20 and, accordingly, controls the DVD data consumption rate, as is described below.

Whenever SD 26 extracts the pack header 201 of a data pack 200, it instructs timer 30 to store the current time in register 56. SD 26 stores the clock reference from the pack header in register 56. Therefore, register 56 stores both the clock reference of the data pack 200 as well as the time at which the pack arrives at SD 26. CPU 54 has access to register 56 and can read its content to determine the difference between the two timing data. In a steady state, when the system is fully synchronized, a fixed timing difference exists between the clock reference of data pack 200 and the time which data pack 200 arrives at SD 26.

Furthermore, after demultiplexing and depacketizing a DVD data pack 200, SD 26 extracts the associated time stamps of the data packet 206 (i.e. PTS 212 and DTS 210), stores the data in FIFO 48 and, accordingly, generates a tag which contains the DTS and

the address of the stored data in FIFO 48. In other words, for each data pack 200, SD 26 generates a tag containing information about both the decode time stamp as well as the address of the stored data in FIFO 48. The generation of the tags is a significant advantage of decoder 20.

The tags thus generated are made available to CPU 54 for further synchronization and control of decoder 20. Using the information stored in a tag, CPU 54 generates a Task Definition Packet (TDP) containing instructions to video decoder 36 about the location of the data in FIFO 48. Upon the arrival of the next Vsync signal, video decoder 36 decodes the data identified by the TDP. If no TDP has been generated by CPU 54, video decoder 36 does not decode any data at the occurrence of the Vsync signal.

Fig. 3 shows the basic format of a TDP. Each TDP is composed of 40 words (each word consists of 32 bits) of which the last 5 words are reserved. The first word of each TDP contains the address (with respect to buffer 48) of the next video picture to be decoded.

Depending on the data consumption rate, CPU 54 may instruct video decoder 36 to decode data out of sequence. Therefore, if a faster data consumption rate is necessary, CPU 54 skips over the next frame of data in FIFO 48 and thus issues a TDP pointing to the address of the succeeding data frame in FIFO 48. If a slower data consumption rate is necessary, CPU 54 does not issue a TDP and thus the previously decoded data frame is displayed.

Since the decision regarding when and which data to decode next is made by CPU 54 and not by video decoder 36, DVD/DVB decoder 20 provides several improvements over those of the prior art. First, video decoder 36 benefits from a relatively simple design because it needs to decode data only when so instructed. Second, because CPU 54 is controlled through software, CPU 54 lends itself to a highly low-level control, thereby adding to the overall flexibility of the system while making the system easily upgradeable. Third, the generation of the tags relives CPU 54 from the computationally intensive and hence costly task of reading the data that flows between various components of decoder 20. The tags enable the CPU to receive the information it needs to determine which data must be decoded, thereby greatly easing the computational burden on the CPU.

The tags generated by SD 26 are read by CPU 54 only when a Vsync signal arrives. Therefore, in a steady state and under most operating conditions, CPU 54 is interrupted only when a tag is present at the occurrence of a subsequent Vsync signal; this is an advantage of the DVD/DVB decoder 20 which generates fewer interrupts than do other known systems. Moreover, because in the steady state, the interrupt requests are only generated during Vsync occurrences, CPU 54 advantageously has an entire Vsync time period (i.e. the time interval between two successive Vsync signals) to service those interrupt. In particular, when a Vsync signal arrives, CPU reads FIFO 48 to determine whether

SD 26 has generated any tags. If one or more tags exist, CPU performs its assigned tasks and generates corresponding TDPs. However, CPU 54 has the entire period from the reading of the tags until the arrival of the next Vsync signal to complete all of its tasks including the generation of the TDPs. Hence, CPU 54 may continue to finish its ongoing tasks uninterrupted, provided, however, that CPU 54 services the requested interrupt prior to the arrival of the next Vsync signal. Therefore, decoder 20 benefits from very relaxed timing requirements. The relaxed timing requirements resulting from the availability of an entire Vsync period to CPU 54 to read the tags and to generate corresponding TDPs is a significant advantage of DVD decoder 20 over the known DVD decoders which because they require their CPU control units (e.g. CPU) to immediately--but temporarily--abandon their ongoing tasks to service the interrupts in a very short time period, impose tight and rigid system timing requirements.

As stated above, although in a steady state and during normal operating conditions, the occurrence of the Vsync signal is the main synchronizing mechanism for generating interrupt requests to CPU 54, other mechanisms such as polling may also be used to prompt the CPU to read the tags and generate TDPs.

The data decoded by video decoder 36 is stored in buffer 50 for subsequent processing by VOP 40. Because CPU 54 decides when and which data is to be decoded next, it also knows whether any decoded data has been

placed in buffer 50. Consequently, when such data exists, at the occurrence of the next Vsync signal, CPU 54 instructs VOP 40 to fetch the data from the buffer for further processing. After a known time period, the data fetched and processed by VOP 40 is displayed on monitor 42.

The generation and processing of the TDPs by CPU 54 and video decoder 36 are pipelined. In particular, in a steady state, CPU 54 generates TDPs that will be processed by video decoder 36 during the next Vsync signal cycle. Therefore, CPU 54 is always generating TDPs one Vsync signal cycle ahead. The pipelining provides CPU 54 and video decoder 36 with an entire Vsync signal cycle to respectively process the TDPs and decode the data, while simultaneously reducing the idle time of the decoder and thereby increasing the utilization and throughput of the decoder.

The processing of audio data when decoder 20 is in the DVD or DVB mode is described next. Referring to Fig. 1, DVD-DSP interface 24 retrieves DVD data from DVD-DSP 46, subsequently reformats this data as a stream of bytes, and supplies the stream of bytes to SD 26. Similarly, Network Port 22 retrieves DVB data from Network Interface 52, reformats this data as a stream of bytes, and supplies the stream of bytes to SD 26. SD 26 extracts the audio data stream from the incoming data, depacketizes the data, and stores the data for storage in buffer 48. Audio decoder 34 decodes the stored audio data and writes the decoded data to data buffer 50. Thereafter, AOP 38 fetches the data from

buffer 50, processes the data and supplies it to audio Digital-to-Analog Converter (DAC) 44. Accordingly, in this embodiment, the audio path includes DVD-DSP interface 24, SD 26, timer 30, audio decoder 34, AOP 38 and CPU 54.

In MPEG, an audio time stamp is located at the beginning of a PES packet, but it actually refers to the first byte of the first audio sync frame that begins in the packet, and thus, there is no alignment between audio sync frames and PES packet boundaries. Unlike the video bit streams, the audio bit streams do not contain unique and identifiable start code patterns (sync words), making it difficult to detect the start of an audio data frame.

In MPEG, a new audio frame is identified by identifying an audio marker that constitutes a repeating 12-bit field. Thus, for example, if the 12-bit pattern of an audio marker is identified three times, then it may be safely assumed, with a high degree of certainty that the frame is an audio data frame.

After depacketizing an audio frame, SD 26 stores the PES payload in buffer 48 and generates a tag informing CPU 54 of the location of the stored audio frame in the buffer. Consequently, SD 26 has knowledge of both the location of the audio frame in the buffer as well as its corresponding time stamp. SD 26, however, does not know the offset between the beginning of the packet and that of the frame.

Audio decoder 34 decodes the data and upon detecting the corresponding sync word, marks the address and so informs the CPU 54. Figure 4 shows the format of the message generated by audio decoder 34 to CPU 54 when audio decoder 34 finds a sync word in the compressed audio stream. Each such message has 32 bytes. Bits [31:20] of each word are reserved, and only the lower 20 bits are used. The various parameters of the message when decoding MPEG audio are defined below:

NCHANS	number of audio channels in the input bit-stream (including LFE)
SAMPRATE	sampling rate
LAYER	1=MPEG layer 1, 2=MPEG layer 2
INBUF_LEVEL	Level (count) of the compressed audio data input buffer
BITRATE	Bit rate in kilobits/sec
FRM_SIZE	Size of the last decoded frame
OUTBUF_WR_PTR	Current write pointer of the output buffer used to store decoded audio data
INBUF_RD_PTR	Current read pointer of the input buffer used to store compressed audio data
FRM_NUMBER	A running count of the frames decoded
STATUS	A status word used for debug and diagnostics
MSG_ID	Linearly incrementing number which acts as an ID for the message
MSG_TYPE	Has the value 3 indicating it is a Sync Found message.

CPU 54 using the marked address and the SD-generated tags establishes correlation between a sync word and its associated audio frame in the buffer. Consequently, CPU 54 uses the information provided by SD 26 and audio decoder 34 to determine when a particular frame of audio data must be presented, the details of which are described below.

Fig. 5 shows an audio buffer 100 in accordance with one embodiment of the present invention. In audio buffer 100, pointer 102 shows where the address part of SD 26 tag points.

The audio decoder 34 of Fig. 1 supplies the location of the audio sync word in system memory (e.g., in the DRAM). When audio decoder 34 detects a sync word, it captures the current system memory address from which it is reading. Sync frames are generally of a fixed size as indicated by audio buffer 100 of Fig. 5. Thus, if CPU 54 knows the start address, it can obtain other start addresses by simply adding a multiple of the sync frame size.

Further, CPU 54 knows the nominal compressed audio data rate. Thus, once CPU 54 has a time stamp, it can add a time offset per byte consumed by AOP 38 to obtain the desired audio decode time. By comparing the desired decode time with the actual system time, CPU 54 determines whether audio decoder 34 is ahead of or behind schedule. CPU 54 can then instruct audio decoder 34 to skip or repeat data as appropriate.

Referring to Fig. 1, timer 30 may perform timer

operations for the audio data processing that are similar to its timer operations for the video data processing, as described above.

In some embodiments, audio decoder 34 is controlled by CPU 54. Under CPU 54 control, audio decoder 34 may operate as a slave co-processor or as a free-running decode engine. In slave co-processor mode, audio decoder 34 decodes one or more audio frames in response to a CPU 54 decode command. A new command is required for every frame (or a number of frames) to be decoded. Upon completion of its tasks, Audio decoder 34 notifies CPU 54 of the completion of its activities by storing a message in the message queue of buffer 48. In some embodiments, audio decoder 34 generates an interrupt request to CPU 54 after completing its tasks. Thereafter, audio decoder 34 remains idle until it receives another command.

In free-running mode, audio decoder 34 decodes audio frames when the amount of data in the input buffer reaches a predetermined threshold level. In this mode, CPU 54 may only generate commands to synchronize or adjust the data consumption rate. Thus, the audio decode rate is regulated by the levels of the input buffer and the output buffer, and the audio decode operation will be suspended while the input buffer content level is too low or the output buffer level is too high relative to predetermined threshold levels.

In addition to data processing, audio decoder 34 maintains audio synchronization by providing CPU 54 with the current positions of the input and output

buffer pointers at the beginning of each audio frame. Based on this information, CPU 54 correlates the encoded and actual presentation time of the audio data and provides audio synchronization adjustment commands as appropriate.

Start-up processing typically occurs on a reset, power-on, or while switching programs. CPU 54 mutes the audio output until audio presentation time stamps arrive in the data stream, and the audio synchronization software is running.

CPU 54 manages the total delay in the audio decode-presentation path by manipulating the depth of the audio input and output buffers. In particular, CPU 54 may control the buffer depth by controlling when audio decode tasks are issued and started. Audio decoder 34, in free-running mode, holds off from processing the next audio frame until the input buffer level reaches a predetermined threshold level (e.g., as provided by CPU 54). The audio decode-presentation path delay remains constant and matches the video decode-presentation path delay.

In steady state mode, minor synchronization adjustments may be required at various times. Due to the nature of audio decoding, there are three different granularities of delay adjustment including, adding or dropping entire audio frames, adding or dropping multiples of thirty-two samples from a frame, and delay adjustments made by changing the audio DAC clock (not shown) for controlled periods of time, a "micro-stepper" approach. Because the audio DAC clock tracks

the transport stream, adjustment of the audio decode loop should be fairly infrequent.

The audio decoding operation also supports bit stream error handling. In particular, the audio decoder 34 detects the bit stream errors by means of a CRC (Circular Redundancy Check) provided in the audio stream. When an error is detected, error concealment is applied to minimize the audible effect of the error. AOP 38 reads the decoded audio data, further processes the data and subsequently supplies the data to DAC 44.

In some embodiments, AOP 38 interfaces to commercially available two-channel and six-channel audio DACs. AOP 38 retrieves data from buffer 50 and transmits the data to external audio DAC 44 at a rate dictated by the DAC clock. Based on a write request from audio decoder 34 and a read request from AOP 38, the MMU (memory management unit) (discussed below with respect to Fig. 6) maintains a read pointer and a write pointer for the audio output buffer in buffer 50. By monitoring the buffer depth, rate of change, and direction of change, CPU 54 determines whether the DAC clock should be increased or decreased in frequency relative to the video (i.e., pixel) clock (not shown).

Referring to Fig. 1, decoder 20 of Fig. 1 also provides a digital video broadcast (DVB) operation. In DVB mode (of operation), decoder 20 of Fig. 1 receives a DVB data stream from the external demodulator, selects the proper video and audio programs, extracts timing and compressed video and audio data, decodes the data, and presents the data via NTSC/PAL TV monitor 42

and audio DAC 44. Similar to the DVD operation as described above with respect to Fig. 1, decoder 20 in DVB mode extracts system information from the DVB data stream and places the system information in buffer 48 for use by CPU 54.

In particular, the DVB video path operation is similar to the DVD video path described above with the following exceptions. In the DVB video path, DVB data arrives at Network Port (NP) 22 from a network interface 52 at 40 Mb/s (megabits per second) instead of at DVD-DSP interface 24 from DVD-DSP 46 at 11 Mb/s. Also, in the DVB video path operation, SD 26 may process a transport stream instead of a program stream. In addition, in the DVB video path operation, VOP 40 does not support sub-pictures.

In particular, NP 22 receives data from the external demodulator network interface 52 and reformats the data as a series of bytes. NP 22 then sends the data to SD 26.

In the video path of the DVB mode, SD 26 operation is generally similar to the DVD mode at the PES packet level and elementary stream level. However, at the transport packet level, SD 26 operation differs from the DVD mode of operation as described below. SD 26 looks for periodic repetitions of the sync field in transport packets and achieves byte alignment by synchronizing itself to the occurrences of packets in the transport packet layer. Also, SD 26 demultiplexes transport packets based on the PID they contain. SD 26 then depacketizes the transport packets into a stream

of PES packets, extracting control and navigational information. In particular, for the video path, the splice information contained in the transport adaptation field is extracted. CPU 54 receives this extracted information, through the messaging system as described above, and uses the information to determine when to force timer 30 to reset the local time to match the input stream clock reference. In one embodiment, the navigational information appears in the form of SI table sections which SD 26 places into buffer 48 for use by CPU 54, as discussed further below with respect to Fig. 7.

In DVB mode, the audio path operation is similar to the video path operation except that audio decoder 34 and AOP 38 replace video decoder 36 and VOP 40, respectively. Also, the operation of audio decoder 34 and AOP 38 is generally similar in the DVD mode and the DVB mode.

Finally, in the DVB mode, the DVB clock control path operation is similar to the DVD clock control path operation described above with the following exceptions described below. First, data comes through NP 22 instead of DVD-DSP interface 24. Second, SD 26 processes a transport stream rather than a program stream. SD 26 extracts a clock reference from the incoming stream as similarly described above for the DVD operation. However, in DVB mode, there can be several clock references. Thus, SD 26 only extracts program clock references (PCR) from transport packets that have a certain PID. In particular, even though SD

26 may demultiplex and depacketize other PID packets with PCRs, only the programmed PID will be used for clock reference operations. Thus, the actual operations with clock reference are similar to as described above in the DVD mode. Unlike the DVD mode in which the data stream is pulled into the system, in the DVB mode, the clock frequency is adjusted to match the data rate of the incoming data stream.

In one embodiment, a DVD/DVB decoder system that includes decoder 20 of the present invention may be implemented using an ASIC (application specific integrated circuit), an on-chip processor (for CPU 54), 2 or 4 megabytes of conventional 16 megabit (MB) SDRAM arranged as 2x512Kx16, expandable to 16MB, a ROM (0.5 to 16 MB) for operating system, application, driver, and diagnostic instruction and data storage, a stereo audio DAC or 6 channel DAC output and, or S/PDIF output for AC-3 playback, and an 8052 microcontroller for front panel, infrared (IR) and general purpose input/output (I/O).

Fig. 6 shows SD 26 connections to other modules in decoder 20 of Fig. 1 in accordance with one embodiment of the present invention. In particular, SD 26 connects to timer 30, a memory management unit (MMU) 50, a host bus interface 52, and a network port/DVD controller 54.

Referring to Fig. 6, as part of depacketizing and transporting the byte stream, SD 26 writes the extracted information to MMU 50. MMU 50 subsequently transfers the information into the system memory (e.g.,

buffer 48 of Fig. 1). Also, the audio decoder 34 of Fig. 1 and the video decoder 36 of Fig. 1 request audio and video data, respectively, from the system memory (e.g., the buffer 48 of Fig. 1) using MMU 50. In one embodiment, the system memory includes thirty-two separate buffers (e.g., buffer 48 includes thirty-two separate sub-buffers): one buffer each for MPEG video, audio, sub-picture, teletext, and various other data and control streams. Further, SD 26 tags certain events in the bit stream being written to system memory with, for example, the desired decode or presentation time, the current write location in system memory, etc. SD 26 presents the tags to CPU 54 through the message queue (e.g., an event FIFO) stored in buffer 48 of Fig. 1.

For example, when SD 26 detects clock reference fields (CR) in the input stream, SD 26 provides the value to timer 30. Timer 30 may use this data to set the current system time when decoder 20 of Fig. 1 is attempting to gain initial synchronization. When the system (decoder 20) is already synchronized, SD 26 can capture the current time from the timer 30 when SD 26 detects CR fields and store the CR fields in registers for presentation to CPU 54.

SD 26 extracts system time stamps, either PCR for transport streams or SCR for program streams, and SD puts these extracted time stamps in host-accessible registers along with the current system time and the level of each FIFO in the receive packet. Using the time stamp and FIFO level versus system time data as

phase measurements into a software-controlled phase-locked loop, CPU 54 can adjust the frequency of the video pixel clock and the audio oversampling clock so that the system time base matches the source material time base on average. The adjustment will lower the rate of skip/repeat events in the presentation processes. When splices occur in a transport stream, there may be a discontinuity between time stamps on either side of the splice.

For transport streams, SD 26 separates a particular program from the remaining programs in the stream. Generally, a program will consist of at most one video elementary stream, at most one audio elementary stream, possibly a teletext stream or a sub-picture stream, and a number of streams containing system information tables. A stream as defined by a service provider may have more than one elementary video or audio stream, but decoder 20 of Fig. 1 only decodes one of each type of stream at a given time.

SD 26 accepts PES packets, SI or PSI tables, or program stream system headers from NP 2. SD 26 places depacketized data into one of a number of circular buffer tags supplying pointers to units within a buffer. For example, for video data, SD 26 provides pointers to picture start codes. For audio data, there are pointers to packets with time stamps present in the stream. For program streams, there generally will be a single default PID used for routing purposes.

Accordingly, SD 26 of Fig. 7 selectively transports demultiplexed and depacketized byte streams.

For example, if decoder 20 only includes one audio decoder 34 as shown in Fig. 1, then SD 26 only transports audio data for the user programmed language for audio output (e.g., English or Japanese) (assuming that the audio decoder can only decode audio data for one language at a time).

Fig. 7 shows an event tag format in accordance with one embodiment of the present invention. SD 26 generates tags that are stored in buffer 48 of Fig. 1. In another embodiment, SD 26 generates an interrupt after it writes a new tag to MMU 50 of Fig. 6. In particular, the tags contain addresses in the system memory (e.g., in a RAM, DRAM, or SRAM that includes buffer 48 of Fig. 1) where data associated with certain events in the input stream can be found. Thus, Fig. 7 provides a table for an event tag that lists the event tag bit definitions in accordance with one embodiment of the present invention.

Fig. 8 shows a layered-packetized protocol that is demultiplexed, depacketized, and transported by SD 26 of Fig. 1 in accordance with one embodiment of the present invention. In particular, a byte stream 100 represents a layered-packetized protocol that includes PCI packets, DCI packets, video packets, audio packets, and DSI packets. For example, MPEG represents a layered-packetized protocol. In one embodiment, SD 26 of Fig. 1 can handle the transport layer, the PES layer, and the table layer of a layered-packetized protocol such as MPEG2, and CPU 54 can handle parsed elementary stream layers and audio/video streams. A

layered-packetized protocol typically includes header information and payload information in the packets of data, and the header information contains various information respecting the packet or perhaps subsequent or preceding packets of similar or different packet types. In particular, SD 26 can use the header information to selectively demultiplex packets of byte stream 100, as discussed above with respect to Figs. 1 and 5.

Fig. 9 provides a message queue 106 in accordance with one embodiment of the present invention. Message queue 106 includes tags. Message queue 106 is a FIFO queue stored in a sub-buffer of buffer 48 of Fig. 1. In particular, when SD 26, as discussed above with respect to Figs. 1 and 5, identifies significant information such as the location of the start of a new video frame or a presentation time stamp of a particular video frame, then SD 26 generates a tag such as a tag 108 and stores tag 108 in message queue 106. In one embodiment, SD 26 generates a message that audio or video data is stored in buffer 48 of Fig. 1 and ready for decoding even prior to storing all the audio or video data, respectively, in buffer 48. CPU 54 periodically accesses message queue 106 (e.g., every Vsync) and reads the tags in real time.

In particular, message queue 106 provides a FIFO queue such that the most recent event is stored in the tag that is at the top of the FIFO queue. For example, referring to Fig. 8, if video data is identified prior to audio data in byte stream 100, then a tag in message

queue 106 identifying a time stamp for the video data of byte stream 100 would be lower in message queue 106 than a tag identifying a time stamp for the audio data (e.g., such tags may also include an address identifying the location of the audio and video data, respectively, in buffer 48 of Fig. 1).

Accordingly, SD 26 performs time/space multiplexing that allows CPU 54 to function in a batch mode rather than generating interrupts to CPU 54 each time a significant piece of information is identified by SD 26. For example, message queue 106 as used by SD 26 represents an alternative approach to having SD 26 generate interrupts to CPU 54 each time SD 26 identifies significant information (the interrupt approach). The interrupt approach is inefficient, because every time an interrupt is generated for CPU 54, CPU 54 spends a few cycles processing the interrupt. Thus, SD 26 advantageously uses message queue 106 to implement a hardware-based messaging system (hardware-based tagging mechanism).

Moreover, because CPU 54 is processing information one Vsync signal cycle ahead, CPU 54 typically can access the tags in message queue 106 and process the tags appropriately to program audio decoder 34 of Fig. 1 or video decoder 36 of Fig. 1 appropriately in real time. For example, the MPEG protocol assumes that there is at least one frame worth of delay such that the presentation time stamps allow for such delay upon receipt of the video data and the audio data. Accordingly, decoder 20 uses SD 26 and message queue

106 to take advantage of this scheme by providing a hardware-based messaging system for decoder 20 of Fig. 1. In one embodiment, CPU 54 uses MMU 50 of Fig. 6 to request all the messages stored in message queue 106, and CPU 54 performs this task periodically (e.g., every Vsync) such that the tags are processed well in advance as necessary for appropriate display timing. Alternatively, SD 26 may in all or some cases also raise an interrupt to CPU 54 when generating a tag for message queue 106.

Accordingly, SD 26 of Fig. 1 demultiplexes, depacketizes, and transports byte streams and provides a hardware-based tagging mechanism in accordance with one embodiment of the present invention. CPU 54 reads the tags and, based on the tags, programs decoder 20 of Fig. 1 such that the audio data and video data are output within the presentation time requirements.

Although particular embodiments of the present invention have been shown and described, it will be obvious to those skilled in the art that changes and modifications may be made without departing from the present invention in its broader aspects, and therefore, the appending claims are to encompass within their scope all such changes and modifications that fall within the true scope of the present invention.

CLAIMS:

1. A decoder system comprising:

a stream demultiplexer for demultiplexing and depacketizing data bytes and for storing the demultiplexed and depacketized data bytes in a data buffer, said stream demultiplexer further generating messages about the stored data and their location in the data buffer; and

a control unit for receiving the generated messages and for providing in response thereto instructions about the stored data.

2. The decoder system of Claim 1 wherein said data bytes are DVD or DVB data bytes.

3. The decoder system of Claim 2 wherein the messages generated by the stream demultiplexer about the audio and the video components of a DVD or DVB data byte are recorded on tags containing information about the time stamp of the data and their storage location in the data buffer.

4. The decoder system of Claim 3 wherein in response to a video tag, said control unit generates a task definition packet specifying the location of the video data stored in the data buffer.

5. The decoder system of Claim 4 wherein in response to a task definition packet, a video

decoder of the decoder system fetches the video data from the data buffer and decodes it at the specified time.

6. The decoder system of Claim 5 wherein said control unit responds to a video tag during the intervals between occurrences of a synchronization signal.

7. The decoder system of Claim 6 wherein said video decoder fetches the video data from the data buffer and decodes it at the specified time during the intervals between occurrences of the synchronization signal.

8. The decoder system of Claim 7 wherein during each synchronization cycle, said control unit generates task definition packets for decode by the video decoder during the next synchronization cycle, said synchronization cycle defined as the time period between two successive synchronization signals.

9. The decoder of Claim 8 wherein in a steady state and during the normal operating conditions of the decoder system, said control unit is interrupted only during the occurrence of a synchronization signal for audio and video decode and presentation.

10. The decoder of Claim 9 wherein said video decoder fetches and decodes data only in response to the existence of a task definition packet.

11. The decoder of Claim 10 wherein said control unit comprises a central processing unit.

12. The decoder of Claim 11 further comprising:

an audio decoder for retrieving audio data stored in the data buffer and for decoding the retrieved audio data; and

a video decoder for retrieving video data stored in the data buffer and for decoding the retrieved video data.

13. The decoder of Claim 12 wherein said audio decoder detects the occurrence of a sync word in an audio data frame.

14. The decoder of Claim 13 wherein said central processing unit determines the presentation time of an audio data frame using the time stamp of the associated data packet extracted by the stream demultiplexer and the sync word detected by the audio decoder.

15. The decoder of Claim 14 further comprising a set of data buffers coupled to the audio decoder and the video decoder and comprising audio output buffer and video frame stores.

16. The decoder of Claim 15 further comprising:

an audio output processor coupled to the audio output buffer for retrieving the decoded audio data and for processing thereof; and

a video output processor coupled to the video frame stores for retrieving the decoded video data and for processing thereof.

17. The decoder of Claim 16 further comprising:

an audio digital-to-analog converter coupled to the audio output processor for converting the processed digital data to analog data; and

a video display coupled to the video output processor for displaying the processed video data.

18. The decoder of Claim 17 further comprising a DVD-DSP interface coupled to the stream demultiplexer, the DVD-DSP interface receiving a DVD bit stream, and the DVD-DSP interface transmitting a DVD byte stream to the stream demultiplexer.

19. The decoder of Claim 18 further comprising a network port coupled to the stream demultiplexer, the network port receiving a DVB bit stream, and the network port transmitting a DVB byte stream to the stream demultiplexer.

20. The decoder of Claim 19 further comprising:

a timer for maintaining local current time; and

a clock generator coupled to the timer for maintaining clock references for the decoder.

21. The decoder of Claim 20 wherein the first buffer comprises a message queue for storing messages from the stream demultiplexer for the central processing unit.

22. The decoder of Claim 20 wherein the first buffer comprises a video buffer, an audio buffer, a control data buffer and a queue of the stream demultiplexer tags, each stream demultiplexer tag comprising a pointer to a video start code in the video buffer or to an audio sync frame in the audio buffer or to a beginning of a packet in the control data buffer.

23. The decoder of Claim 21 wherein the decoder is implemented as an ASIC.

24. A method for decoding data bytes comprising:

demultiplexing, depacketizing, and storing the demultiplexed and depacketized data bytes in a data buffer;

generating messages about the stored data bytes to a control unit; and

generating instructions about the stored data bytes using the control unit.

25. The method of Claim 24 wherein the demultiplexing and depacketizing data bytes comprise demultiplexing and depacketizing DVD or DVB data bytes.

26. The method of Claim 25 wherein the act of generating messages about the stored data bytes to a control unit comprises generating tags containing information about the time stamps of the data and their storage location in the data buffer.

27. The method of Claim 26 further comprising generating a task definition packet in response to the generation of said tag, each task definition packet specifying the location of the stored data.

28. The method of Claim 27 wherein the act of generating a task definition packet occurs during the intervals between occurrences of a synchronization signal.

29. The method of Claim 28 further comprising fetching and decoding the stored data in response to the generation of a task definition packet.

30. The method of Claim 29 wherein the act of fetching and decoding the stored data in response to the generation of a task definition packet occurs during the intervals between occurrences of

a synchronization signal.

31. The method of Claim 30 wherein for each data packet the act of generating a task definition packet occurs one synchronization signal cycle before the act of fetching and decoding, said synchronization cycle defined as the time period between two successive synchronization signals.

32. The method of Claim 31 further comprising generating interrupt requests only during the occurrence of a synchronization signal when said decoder is in a steady state and is operating under normal operating conditions.

33. The method of Claim 32 wherein the act of generating tags to the control unit involves generating tags to a control unit that is a central processing unit.

34. The method of Claim 33 further comprising:

retrieving the stored audio data and decoding thereof using the audio decoder; and

retrieving the stored video data and decoding thereof using the video decoder;

35. The method of Claim 34 further comprising detecting the sync word of an audio data frame using the audio decoder.

36. The method of Claim 35 further comprising determining the presentation time of an audio data frame using the time stamp of the data packet and the sync word of the audio data frame.

37. The method of Claim 36 further comprising storing the decoded audio data and the decoded video in a set of data buffer.

38. The method of Claim 37 further comprising:

retrieving the decoded audio data from the set of data buffers for processing and supplying the processed data to an audio digital-to-analog converter; and

retrieving the decoded video data from the second buffer for processing and supplying the processed data to a video display.

Sprint Exhibit 1006 - Page 36 of 50

DVD/DVB DECODER

Christopher K. Wolf
Ygal Arbel
Himanshu A. Sanghavi

ABSTRACT OF THE DISCLOSURE

The decoder, in accordance with the present invention, includes a stream demultiplexer that demultiplexes and depacketizes a stream of DVD or DVB data packets, stores the demultiplexed and depacketized data in a data buffer and subsequently generates a tag specifying the time stamp of the data and the location of the stored data in the buffer. The CPU, using the information generated for each tag, generates a task definition packet instructing the audio and the video decoders about the location and the time that a particular data in the buffer must be decoded, simplifying the operation and design of the decoders. The generation of the task definition packets and the decoding of the data are synchronized with respect to a synchronizing signal generated within the decoder. Because the CPU is not required to demultiplex and depacketize the data, it need not have high processing power and is thus relatively inexpensive. Under the normal operating conditions, the CPU is interrupted only during the occurrence of the synchronization signals. The generation of the

task definition packets and the decoding of the data is pipelined. Therefore, during each synchronization signal, the decoder decodes a data packet in accordance with a task definition packet that the CPU generated during the previous synchronization signal. The audio decoder detects the audio sync word and so informs the CPU which along with the time stamp of the data provided thereto by the stream demultiplexer determines the presentation time of the audio frame.

DOCKET NO. 50004-0008

6682h0 86h0260

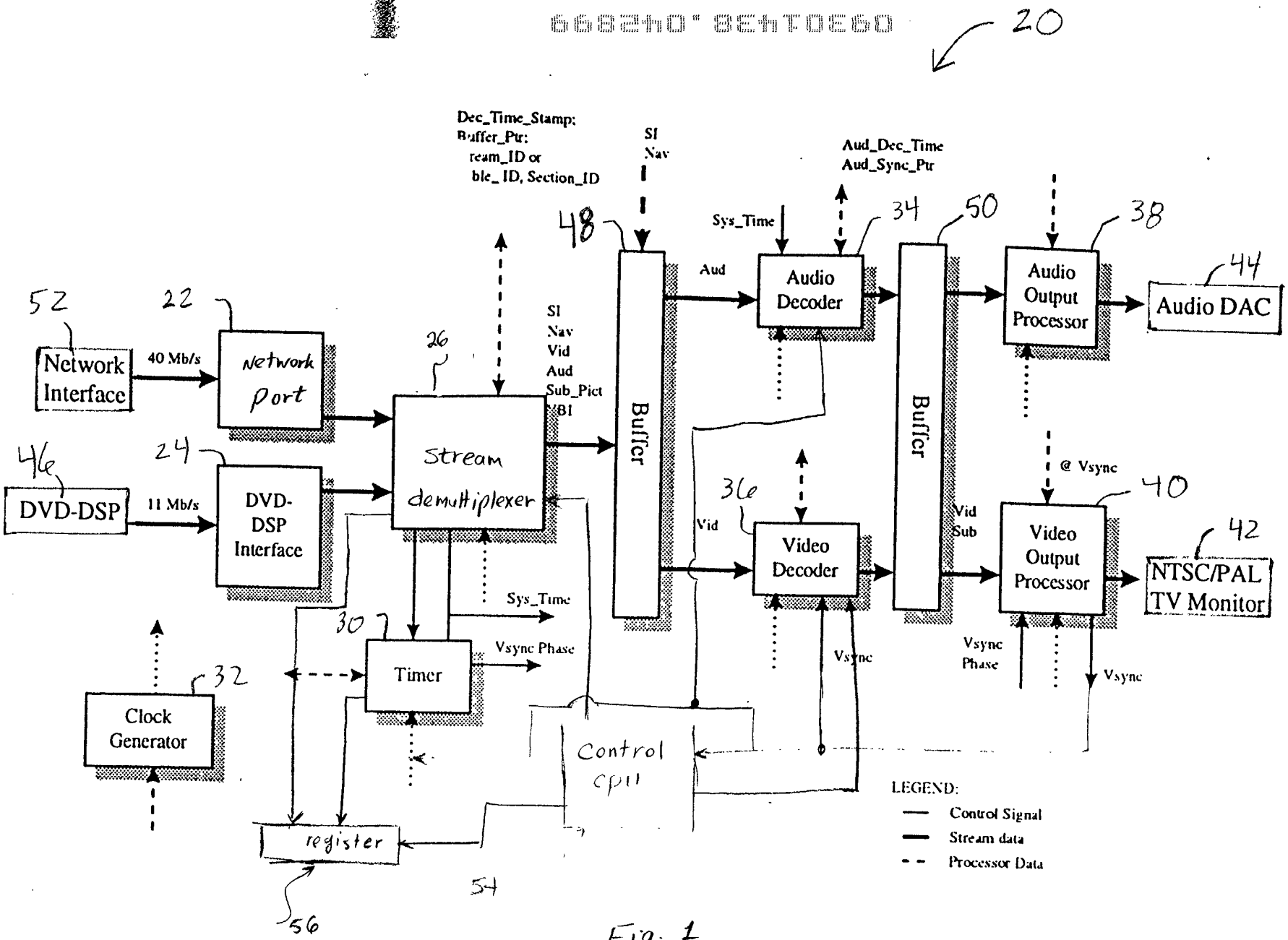


Fig. 1

Fig. 2A

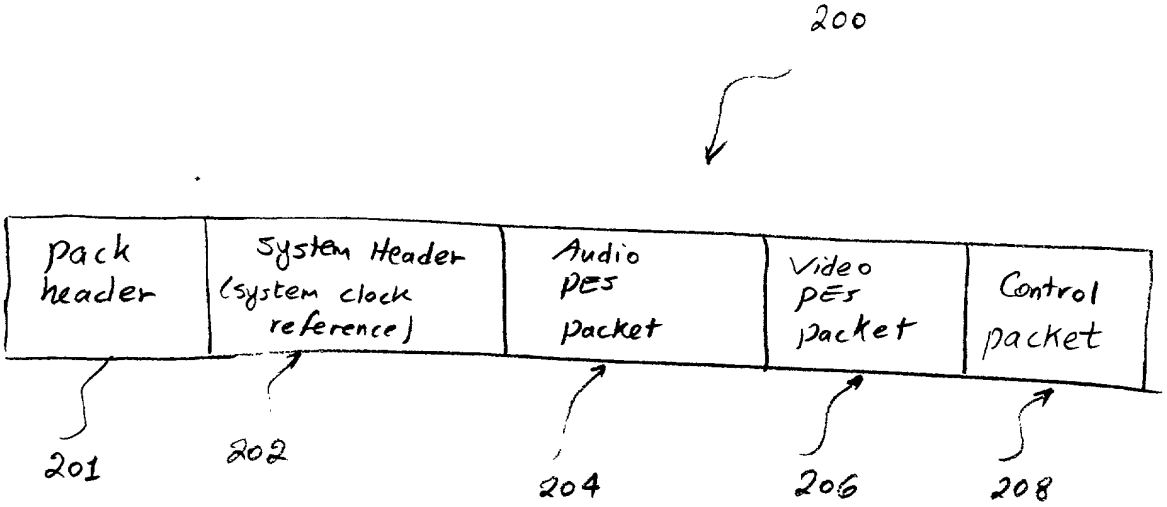


Fig. 2B

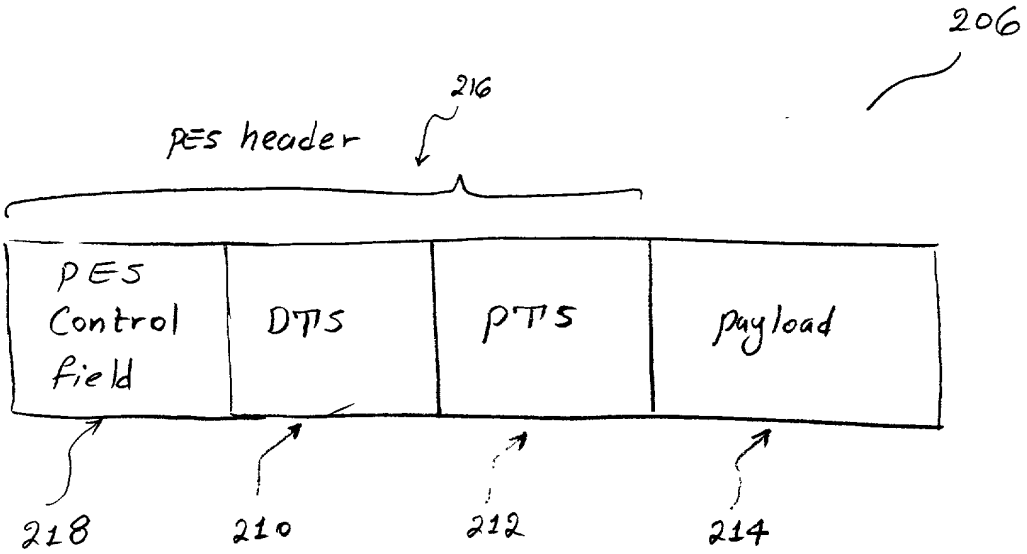
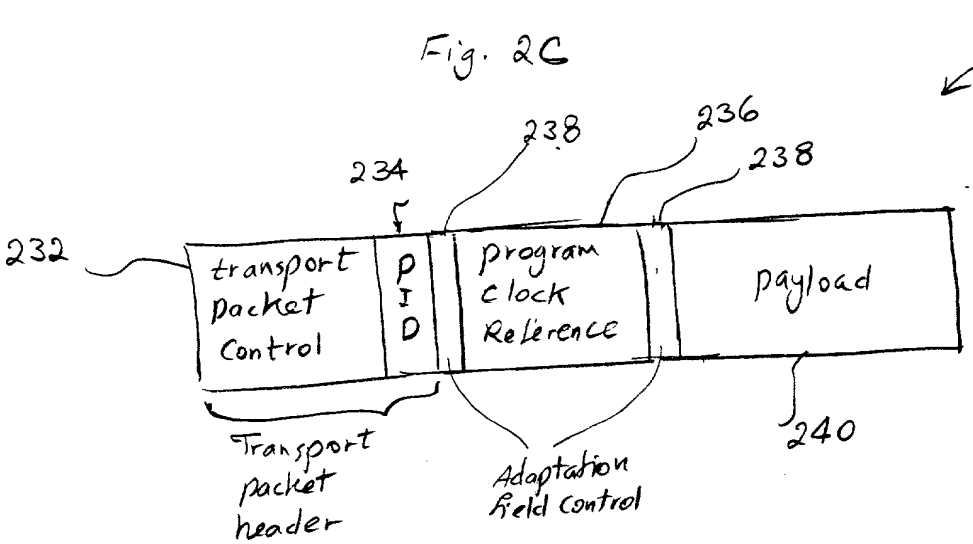


Fig. 2C



Format of a Task Definition Packet

Word offset	Data [31:0]
0	Address of picture to be decoded
1	picture size bits
2	picture Control and Fcode Register bits
3-34	Quantizer Matrix Coefficients
35-39	Reserved

Fig. 3

Sony Exhibit 1008 - Page 0005 of 0005

Audio Sync Word Message Format

bits [31:20]	bits [19:10]	bits [9:0]
	SAMPRATE	NCHANS
	INBUF_LEVEL	LAYER
	FRM_SIZE	BITRATE
	OUTBUF_WR_PTR	
	INBUF_RD_PTR	
	FRM_NUMBER	
	STATUS	
	MSG_TYPE = 3	MSG_ID = 0x100 - 0x1FF

Fig. 4

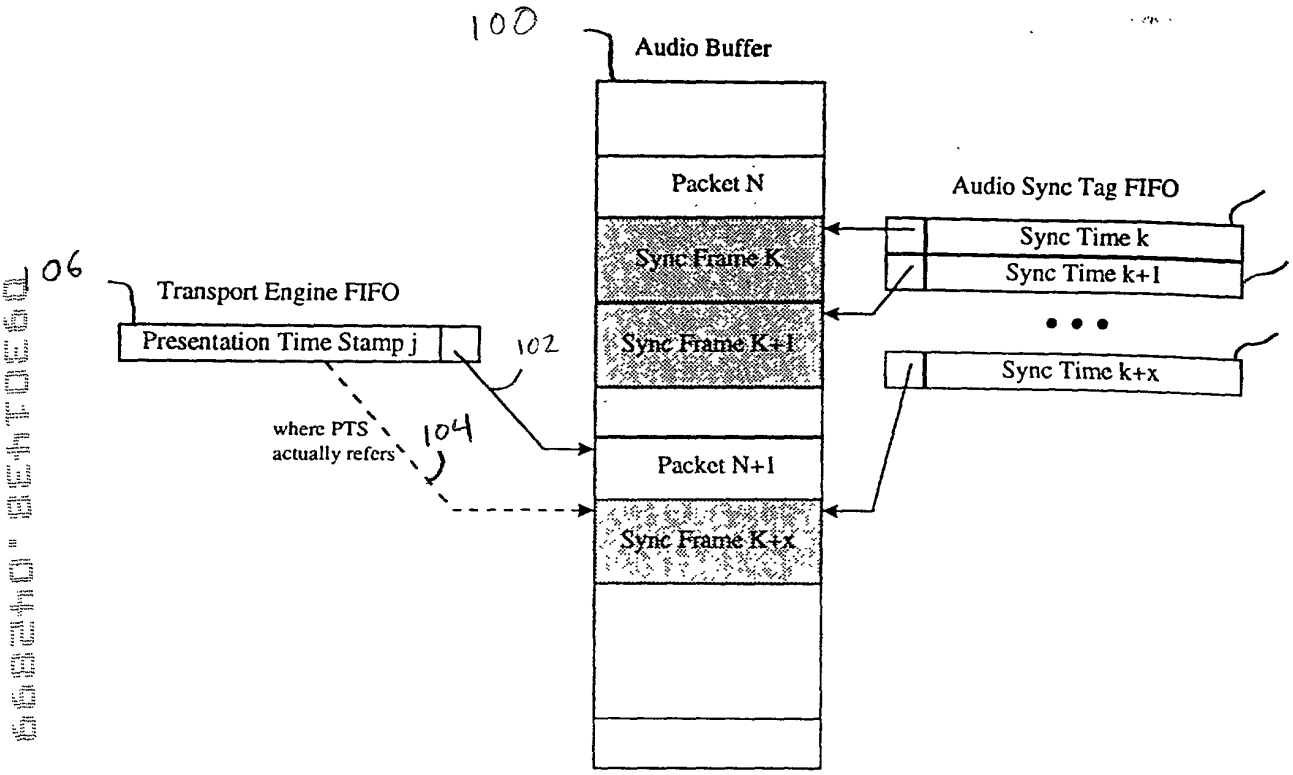

100
Audio Buffer
Packet N
Sync Frame K
Sync Frame K+1
Packet N+1
Sync Frame K+x
Audio Sync Tag FIFO
Sync Time k
Sync Time k+1
Sync Time k+x
106
Transport Engine FIFO
Presentation Time Stamp j
102
where PTS
actually refers
104

Fig. 5

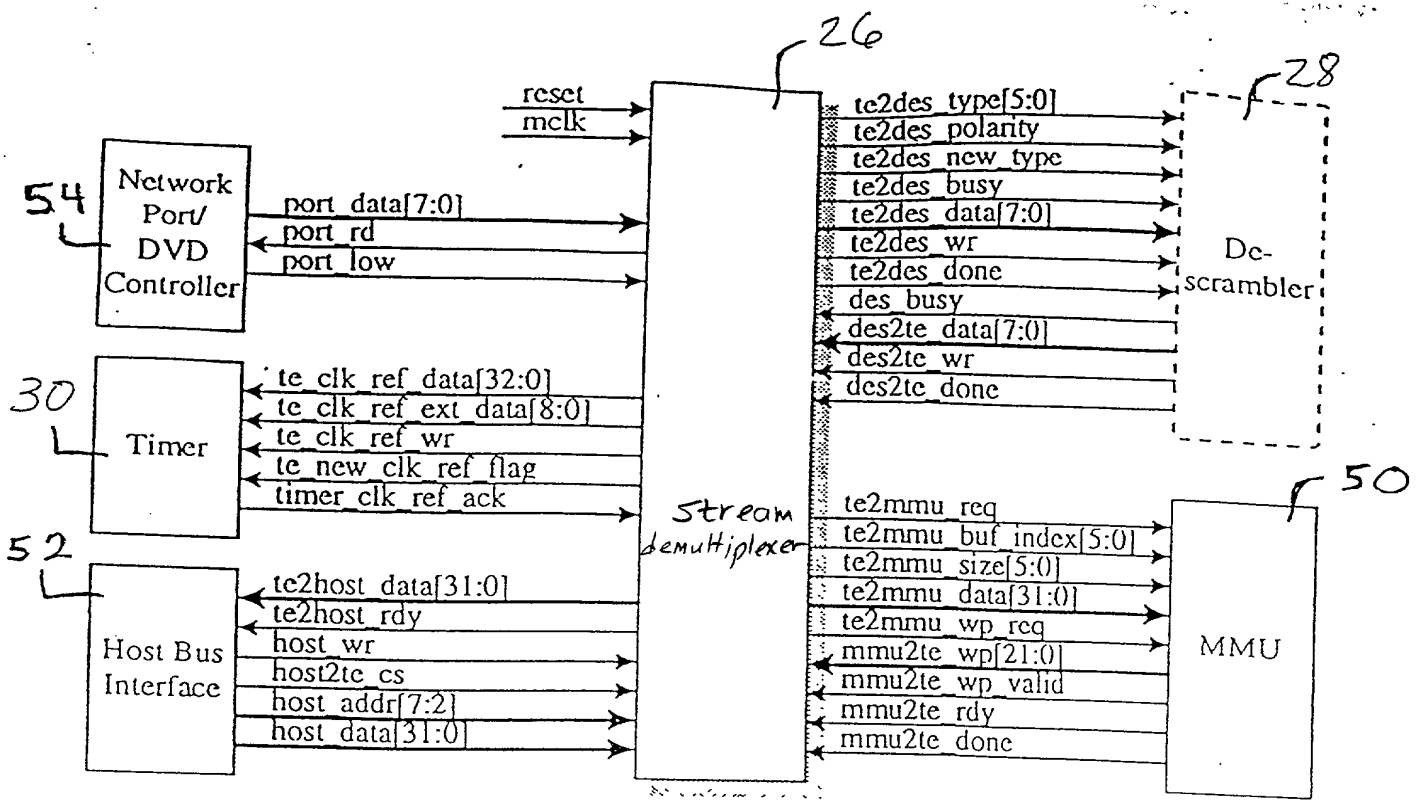


Fig. 6

Docket No. 50277-0005

t				
N	31:6	25		Reserved
	6:5	2	DTS	Valid DTS present 00 = no DTS 01 = DTS carried in PES 10 = DTS carried in DTS_next_AU field of transport packet 11 = reserved
	4:3	2	ERROR	Error Type 00, 01 = no error 10 = Transport Propagated Error 11 = Transport Continuity Error
	2:0	3	EVENT	Event Type 000 =Video Start Code 001 = Packet 010 = Substream Section 011 = Table Section 100-110 = reserved 111 = mid-stream error
N+1	31:27	5	BUFFER	The number of the buffer containing the event
	26:22	5	OFFSET	Offset from address of the event, modulo buffer length
	21:0	22	ADDRESS	The byte address of the event in the buffer, minus offset modulo buffer length
N+2	31:0	32	DATA_LO	Event Data Low. If EVENT is 000 - 10 : If DTS is 0 : bits [31:0] = reserved 1 : bits [31:0] = DTS[32:1] 011 :bits [31:0] = table header bytes 1, 2, 3, 4 100 - 111 : reserved
N+3	31 0	32	DATA_HI	Event Data High. If EVENT is 000 - 01 : bits [30.0] = reserved. If DTS is 0 bits [31] = reserved 1 bits [31] = DTS[0]
				010 bits [23 0] = sub-stream header bytes 2, 3, 4 if Substream ID TYPE = 01 or 10 bits [7 1] = reserved If DTS is 0 bit [31] = reserved 1 bit [31] = DTS[0] 011 bits [31 0] = table header bytes 5, 6, 7, 8 100 - 111 reserved

Fig. 7

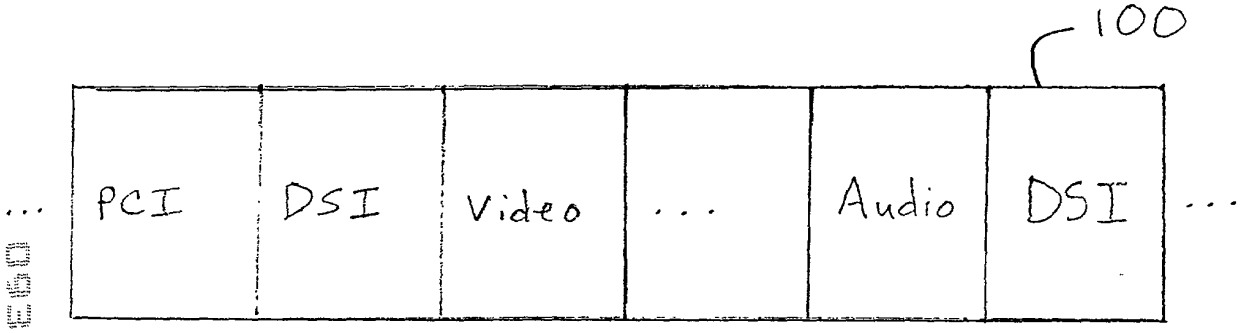


Fig. 8

Sony Ex. 1003 - Page 8 of 13

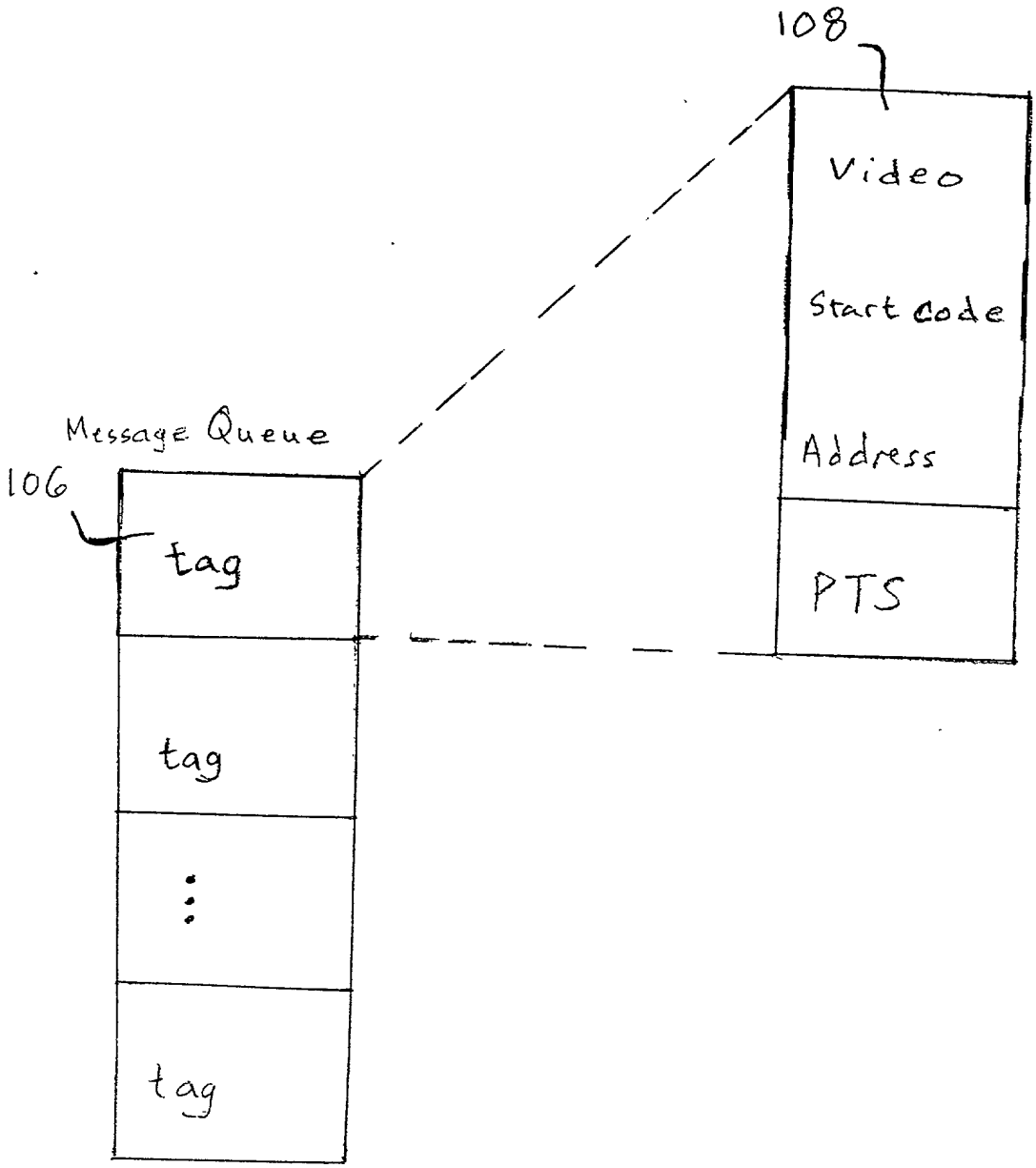

108
Video
Start Code
Address
PTS
Message Queue
106
tag
tag
tag

Fig. 9

DECLARATION FOR PATENT APPLICATION AND POWER OF ATTORNEY

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below adjacent to my name.
I believe I am the original, first and joint inventor of subject matter (process, machine, manufacture, or composition of matter, or an improvement thereof) which is claimed and for which a patent is sought by way of the application entitled

DVD/DVB Decoder

which (check) ☒ is attached hereto.
☐ and is amended by the Preliminary Amendment attached hereto.
☐ was filed on _____ as Application Serial No. _____
☐ and was amended on ___ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information, which is material to patentability as defined in Title 37, Code of Federal Regulations, § 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, § 119(a)-(d) of any foreign application(s) for patent or inventor's certificate or any PCT international application(s) designating at least one country other than the United States of America listed below and have also identified below any foreign application(s) for patent or inventor's certificate or any PCT international application(s) designating at least one country other than the United States of America filed by me on the same subject matter having a filing date before that of the application(s) of which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
Number	Country	Day/Month/Year Filed	Yes	No
N/A			☐	☐

I hereby claim the benefit under Title 35, United States Code, § 119(e) of any United States provisional application(s) listed below:

Provisional Application Number	Filing Date
N/A	

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s) or PCT international application(s) designating the United States of America listed below and, insofar as the subject matter of each of the claims of this application is not

disclosed in the prior application(s) in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose information, which is material to patentability as defined in Title 37, Code of Federal Regulations, § 1.56, which became available between the filing date of the prior application(s) and the national or PCT international filing date of this application:

Application Serial No.	Filing Date	Status (patented, pending, abandoned)
N/A		

I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and to transact all business in the United States Patent and Trademark Office connected therewith:

Alan H. MacPherson (24,423); Brian D. Ogonowsky (31,988); David W. Heid (25,875); Norman R. Klivans (33,003); Edward C. Kwok (33,938); David E. Steuber (25,557); Michael Shenker (34,250); Stephen A. Terrile (32,946); Peter H. Kang (40,350); Ronald J. Meetin (29,089); Ken John Koestner (33,004); Omkar K. Suryadevara (36,320); David T. Millers (37,396); Kent B. Chambers (38,839); Michael P. Adams (34,763); Michael J. Halbert (40,633); Gary J. Edwards (41,008); William B. Tiffany (41,347); James E. Parsons (34,691); Daniel P. Stewart (41,332); Philip W. Woo (39,880); John T. Winburn (26,822); Tom Chen (42,406); Fabio E. Marino (43,339); William W. Holloway (26,182); Elaine H. Lo (41,158); Don C. Lawrence (31,975); Marc R. Ascolese (42,268); Carmen C. Cook (42,433); David G. Dolezal (41,711); Michael P. Noonan (42,038); Roberta P. Saxon (43,087); Bernice Chen (42,403); Mary Jo Bertani (42,321); Dale R. Cook (42,434); Sam G. Campbell (42,381); Matthew J. Brigham (44,047); Robert B. Morrill (43,817); Glen B. Choi (43,546); Hugh H. Matsubayashi (43,779); Margaret M. Kelton (44,182); Joseph T. VanLeeuwen (44,383); William C. Cray (27,627); Eugene Conser (39,149); Coleman Reif (38,593); Allen Tremain (40,207); Patrick Duncan (41,721; and Chad R. Walsh (43,235).

Please address all correspondence and telephone calls to:

Edward C. Kwok
Attorney for Applicant(s)
SKJERVEN, MORRILL, MacPHERSON, FRANKLIN & FRIEL LLP
25 Metro Drive, Suite 700
San Jose, California 95110-1349

Telephone: 408-453-9200
Facsimile: 408-453-7979

I declare that all statements made herein of my own knowledge are true, all statements made herein on information and belief are believed to be true, and all statements made herein are made with the knowledge that whoever, in any matter within the jurisdiction of the Patent and Trademark Office, knowingly and willfully falsifies, conceals, or covers up by any trick, scheme, or device a material fact, or makes any false, fictitious or fraudulent statements or representations, or makes or uses any false writing or document knowing the same to contain any false, fictitious or fraudulent statement or entry, shall be subject to the penalties including

fine or imprisonment or both as set forth under 18 U.S.C. 1001, and that violations of this paragraph may jeopardize the validity of the application or this document, or the validity or enforceability of any patent, trademark registration, or certificate resulting therefrom.

Full name of first joint inventor:	CHRISTOPHER K. WOLF		
Inventor's Signature:	Christ K. Wolf	Date:	4/28/99
Residence:	San Jose, California		
Post Office Address:	1649 Salisbury Drive San Jose, California 95124-4730	Citizenship:	USA

Full name of second inventor:	YGAL ARBEL		
Inventor's Signature:	Ygal Arbel	Date:	4/28/99
Residence:	Belmont, California		
Post Office Address:	2809 Monte Cresta Drive Belmont, California 94002	Citizenship:	USA

Full name of third joint inventor:	HIMANSHU A. SANGHAVI		
Inventor's Signature:	Sanghavi	Date:	4/28/99
Residence:	Fremont, California		
Post Office Address:	34750 Siward Drive Fremont, California 94555	Citizenship:	India